IN THE CLAIMS

1. (CURRENTLY AMENDED) A computer-implemented method for improving compression for storage of a plurality of parallel data element sequences comprising:

(a) creating a dictionary of unique values for each of said data element sequences, wherein each dictionary associates ~~contains~~ a numeric index with each unique value,

(b) forming an n-ary tree with leaf and interior nodes wherein:

(1) each said leaf node corresponds to one of said dictionaries,

(2) each said interior node associates a numeric index with tuples of numeric indexes from other subordinate leaf or interior nodes, and

(3) interior nodes are capable of storing one or more sequences of mutually-consecutive tuples by representing said sequences in a form that uses less storage space than representing said sequences as individual tuples, and

(4) one or more interior nodes are capable of:

i. recording the addition of a tuple that extends a tuple run by incrementing the length field of said tuple run, or

ii. recording the addition of a tuple that invalidates an existing tuple run by splitting said tuple run into one or more subruns, wherein none of the tuples of said subruns contain any element of said added tuple, or

iii. recording the addition of a tuple that has not been previously added to said interior node, wherein said added tuple does not extend a tuple run, by adding said tuple to a tuple collection, or

iv. any combination of two or more of i, ii, and iii.

2. (CURRENTLY AMENDED) The method of claim 1, wherein each unique value of a leaf node or each unique tuple of an interior node is capable of being associated with a count of the number of times that value or tuple of values occurred in the parallel data element sequences.

3. (CURRENTLY AMENDED) The method of claim 1, further including a means for efficiently processing a subset of a tree's leaves, comprising the following steps:
   (a) defining a gate field ~~in~~ for one or more interior nodes,
   (b) setting the value of said gate field ~~in~~ for each said interior node, to indicate which of said interior node's branches lead to leaf nodes in said subset,
   (c) following paths that lead to said leaf nodes, and
   (d) processing said leaf nodes encountered.
4. (CANCELED)
5. (PREVIOUSLY PRESENTED) The method of claim 1 further including a method for arranging said n-ary tree comprising the steps of:
   (a) defining a problem space comprising:
      (1) a set of states such that each state contains a set of leaves and zero or more interior nodes, each with two or more other nodes as children,
      (2) a value function, giving a numeric ranking of the value of any state's design,
   (b) defining one or more operators that transform one state to another, and
   (c) searching the problem space, starting from an initial state and applying operators to move to other states until a state with an acceptable n-ary tree design is reached.
6. (CANCELED)
7. (CANCELED)
8. (CANCELED)
9. (CANCELED)
10. (CANCELED)
11. (CANCELED)
12. (CANCELED)
13. (CANCELED)
14. (CANCELED)
15. (CANCELED)

16. (PREVIOUSLY PRESENTED) The method of claim 5, where said method uses an estimate of interior node size, from a function of the sizes of said interior node's child nodes.

17. (CANCELED)

18. (CURRENTLY AMENDED) The method of claim 3, where said processing comprises using values or tokens at said leaves to reconstruct a subset of a stored record ~~further including using the length of at least one of said tuple sequences in the representation of one or more of said tuple sequences~~.

19. (CURRENTLY AMENDED) The method of claim 18, where said reconstructed records are inserted into a new tree~~further including the step of adding one or more of said reconstructed record subsets to another tree~~.

20. (PREVIOUSLY PRESENTED) A computer-implemented method for improving compression for storing a plurality of parallel data element sequences comprising:

(a) creating a dictionary of unique values for each of said data element sequences, wherein each dictionary associates a numeric index with each unique value,

(b) forming one or more n-ary trees with leaf and interior nodes where:

(1) each leaf node is distinct from, and represents a subset of values from one of said dictionaries, and

(2) each interior node associates a numeric index with tuples of numeric indexes from other subordinate leaf or interior nodes.

21. (PREVIOUSLY PRESENTED) The method of claim 20, wherein each unique value of a leaf node and each unique tuple of an interior node is capable of being associated with a count of the number of times that value or tuple of values occurred in the parallel data element sequences.

22. (PREVIOUSLY PRESENTED) The method of claim 20, further including a means for efficiently processing a subset of a tree's leaves, comprising the following steps:

(a) defining a gate field for one or more interior nodes,

(b) setting the value of said gate field for each said interior node, to indicate which of said interior node's branches lead to leaf nodes in said subset,

(c) following paths that lead to said leaf nodes, and

(d) processing said leaf nodes encountered.

23. (PREVIOUSLY PRESENTED) The method of claim 22, where said processing comprises using values or tokens at said leaves to reconstruct a subset of a stored record.

24. (CURRENTLY AMENDED) The method of claim 23, further including the step of adding one or more of said reconstructed record subsets to another tree~~where said processing comprises using values or tokens at said leaves to reconstruct a stored record~~.

25. (PREVIOUSLY PRESENTED) The method of claim 20, further including a method for arranging said n-ary tree comprising:

(a) defining a problem space comprising:

(1) a set of states such that each state contains a set of leaves and zero or more interior nodes, each with two or more other nodes as children,

(2) a value function, giving a numeric ranking of the value of any state's design,

(b) defining one or more operators that transform one state to another, and

(c) searching the problem space, starting from an initial state and applying operators to move to other states until a state with an acceptable design is reached.

26. (PREVIOUSLY PRESENTED) A computer-implemented method for storing a plurality of parallel data element sequences, and efficiently processing elements from a subset of said sequences, comprising:

(a) creating a dictionary of unique values for each of said data element sequences, wherein each dictionary associates a numeric index with each unique value,

(b) forming one or more n-ary trees with leaf and interior nodes where:

(1) each leaf node corresponds to one of said dictionaries,

(2) each interior node associates a numeric index with tuples of numeric indexes from other subordinate leaf or interior nodes,

(3) a gate field is defined for one or more interior nodes,

(c) processing the leaves corresponding to said subset of sequences by:

(1) setting the value of said gate field for each said interior node, to indicate which of said interior node's branches lead to leaf nodes in said subset,

(2) following paths that lead to said leaf nodes, and

(3) processing said elements in said leaf nodes encountered.

27. (CURRENTLY AMENDED) The method in claim 26, where said processing comprises using values or tokens at said leaf nodes to reconstruct a subset of a record.

28. (CURRENTLY AMENDED) The method of claim 27, further including the step of adding one or more of said reconstructed record subsets to another tree~~where said reconstructed records are inserted into a new tree~~.

29. (PREVIOUSLY PRESENTED) The method of claim 26, wherein each unique value of a leaf node or each unique tuple of an interior node is capable of being associated with a count of the number of times that value or tuple of values occurred in the parallel data element sequences.

30. (PREVIOUSLY PRESENTED) The method of claim 26, further including a method for arranging said n-ary tree comprising the steps of:

(a) defining a problem space comprising:

(1) a set of states such that each state contains a set of leaves and zero or more interior nodes, each with two or more other nodes as children,

(2) a value function, giving a numeric ranking of the value of any state's design

(b) defining one or more operators that transform one state to another, and

(c) searching the problem space, starting from an initial state and applying operators to move to other states until a state with an acceptable design is reached.

31. (NEW) The method of claim 30, where said method uses an estimate of interior node size, from a function of the sizes of said interior node's child nodes.

32. (NEW) The method of claim 25, where said method uses an estimate of interior node size, from a function of the sizes of said interior node's child nodes.